

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
|  | | Design Patterns | | | | |  | |
|  |  | | | | | | |  |
|  | | | |  |  | | | |
|  | | | | Mubeen KhalidSP21-BSE-015 |  | | | |
|  | | | | 06.03.2024—LAB TASK 2— |  | | | |
|  | | |  | | |  | | |

#### Answer the following questions.

1. Discuss different uses of **super** keyword in Java.

Ans. In Java **super** keyword is used for the following purposes:

* It is used to refer to the superclass objects.
* It can be used to invoke immediate parent class constructors.
* It can be used to invoke immediate parent class methods.
* It can be used to access members of the superclass that can be overridden in the subclass.
* It can be used to call the superclass’s method within the subclass using super.methodName().

For instance,

//Base class/ parent class

class Vehicle {

    int Speed = 120;

}

// subclass

class Car extends Vehicle {

    int Speed = 180;

    void display() {

        System.out.println("Maximum Speed: "

                + super.Speed);

    }

}

class Lab\_01 {

    public static void main(String[] args) {

        Car small = new Car();

        small.display();

    }

}

1. Is it possible to overload the main method or not.

Ans. Yes, we can overload the main method in Java. But JVM always calls the original method. It does not call the overloaded method.

For instance,

public class MainOverloading {

    // invoked when an int value is passed

    public static void main(Integer args) {

        System.out.println("Overload 1");

    }

    // invoked when a char is passed

    public static void main(char args) {

        System.out.println("Overload 2");

    }

    // Original method

    public static void main(String[] args) {

        System.out.println("Original method");

    }

}

Now, we’ll see that how can we invoke the overloaded method from the original main method().

public class MainMethodOverload2 {

public static void main(String args[])

{

    MainMethodOverload mmOverload= new MainMethodOverload ();

    mmOverload.main();

}

}

1. What are the 6 ways of using ‘this’ keyword in Java.

Ans. Following are the ways of using this keyword:

* It eliminates the confusion between class attributes and parameters.
* It is used to invoke the current class constructor.
* It is used to invoke the current class method.
* It is used to return the current class object.
* It can be used to pass an argument to the method call.
* It can be used to pass an argument to the constructor call.

1. What is marker or tagged interface?

Ans. An interface which contains no methods, fields, or constants in known as marker or tagged interface. It is an empty interface. The information is delivered to both the JVM and compiler.

It has the following syntax:

interface Marker

{

}

1. How to override the static method?

Ans. We cannot override the static method in Java because static methods are bonded at the compile time using static binding.

1. How down casting is possible in Java?

Ans. Down casting means typecasting of a parent object to a child object. It cannot be done implicitly.

It can be done like:

public class Downcasting {

    public static void main(String[] args) {

        // Upcasting

        Parent p = new Child();

        p.name = "downcast";

        // Downcasting Explicitly

        Child c = (Child) p;

        c.id = 1;

        System.out.println(c.name);

        System.out.println(c.id);

        c.method();

    }

}

It must be done externally and by doing this the child class can acquire the properties of the parent class.

1. What are the uses of **final** keyword in Java?

Ans. Final keyword is used to restrict the user. It can be used in many context like:

* Variable
* Method
* Class

Java Final Variable:

If we use final with a variable, then it cannot be changed after. It becomes constant.

class Car {

    final int speedlimit = 90;

    void run() {

        speedlimit = 400;

    }

    public static void main(String args[]) {

        Car C = new Car();

        C.run();

    }

}

Java Final Method:

If a method is made final, then we cannot override it. If we will try to override a method which is final it will give us compile time error.

Java Final Class:

If a class is set to be final, then we cannot extend it. If we will try to extend a final class it will give us compile time error.

1. What is the purpose of private constructor in Java?

Ans. The purpose of private constructor in java is to restrict the creation of the object to implement singleton design pattern.